

S10 C49 - Thinking procedurally

Section 10 - Computational thinking

Monday, October 16, 2023

Key terms

Thinking procedurally

- The task of writing a program a lot simpler by breaking a problem down into smaller parts

Procedural decomposition

- Breaking a problem into a number of sub-problems, so that each sub-problem accomplishes an identifiable task

Modular programming

- Modular design and programming techniques are most useful for large, complex programs

Objectives

- Identify the components of a problem
- Identify the components of a solution to a problem
- Determine the order of the steps needed to solve a problem
- Identify sub-procedures necessary to solve a problem

Thinking procedurally
Unit 10 Computational thinking

Connection

Connection: Thinking procedurally


- Most problems of any size need to be broken down into their component parts
- Think of the problem:
 - “How will I be able to go to University?”
- What are the different aspects of this problem? *which uni? what course? qualifications?*
- Is this a computational problem?
- Suggest other problems that would need to be decomposed into separate, smaller sub-problems

Thinking procedurally
Doing computational thinking

Activation

Decomposition

- Procedural decomposition means breaking a problem into a number of sub-problems, so that each sub-problem accomplishes an identifiable task
- The sub-problems may themselves be further subdivided



Thinking procedurally
Doing computational thinking

Activation

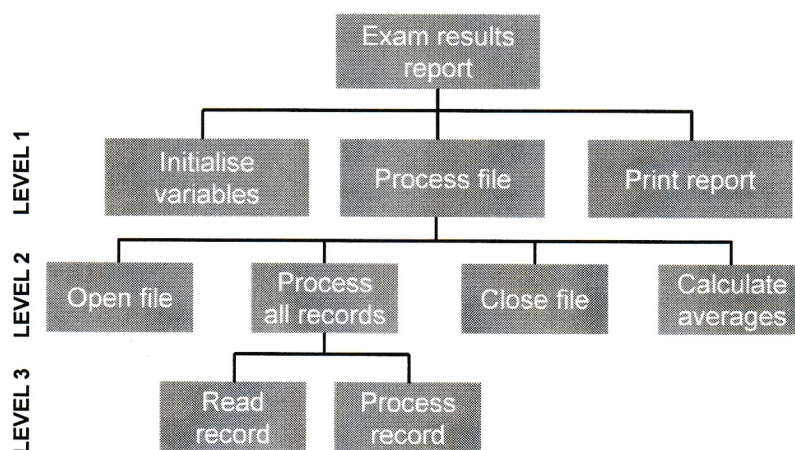
A definition of structured programming

- Structured programming aims to improve the clarity and quality of a program
- It is a method of writing a computer program which uses
 - **Modularization** for program structure and organisation, i.e. breaking the problem down into subroutines
 - **Structured code** for the individual modules – that is, code which uses the basic constructs of **sequence**, **selection** and **iteration**
 - **Recursion**

A top-down design model

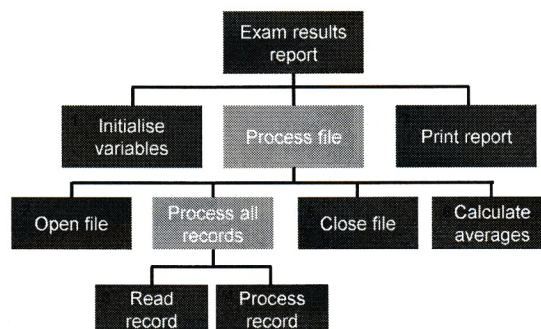
- Structured programming uses a top-down design technique
- A program is divided into sub-procedures, or modules, which are called from the main program
- Any of the sub-procedures may be further broken down into smaller sub-tasks, with the smallest performing a single function
- A hierarchy chart is often used to show the overall program structure

Hierarchy chart



Hierarchy chart

- Each logical process is broken down into smaller components until it cannot be broken down any further
- Execution takes place from left to right, always at the lowest level component
- Selection and iteration are not shown in a hierarchy chart



- Why are two of the boxes shown in a paler colour?

Benefits of modularisation

- Programs are more easily and quickly written
 - Large programs are broken down into subtasks/subroutines that are easier to program and manage
 - Each subroutine (i.e. module) can be individually tested
 - Modules can be re-used several times in a program
 - Frequently used modules can be saved in a library and used by other programs
 - Several programmers can simultaneously work on different modules, shortening development time
- Can you think of some more benefits?

More benefits of modularisation

- Programs are more reliable and have fewer errors
 - It is easier to find errors in small self-contained modules
- Programs take less time to test and debug
- Programs are easier to maintain
 - A well-organised, modular program is easier to follow
 - It is easier to find which module needs to be changed
 - Self-contained modules mean that the change should not affect the rest of the program
 - New features can be added by adding new modules

Q1: Give some other advantages of writing a program as a collection of independent modules.

Good programming practice

- Use meaningful identifiers (e.g. variable and procedure names)
- Define and document the inputs, outputs and preconditions for each sub-procedure
- Add lots of meaningful comments within the program
- At the bottom level, each sub-procedure should perform a single task
- Keep each sub-procedure self-contained by passing parameters and using local variables


Identifying the components

- Once you have identified the component parts of a problem, you can plan the overall method of solution
- This will involve writing procedures or functions and passing parameters
- For example, a car dealer might want a program to help potential customer select options for a new car
- This might use a procedure to display a particular model of a car, passing parameters for colour, number of doors, wheels etc.

Thinking procedurally
Unit 10 Computational thinking

Activation

Choosing car options



Back: Wheels Engine Wheels **Paint** Interior Extras Overview Next: Interior

Black (Non Metallic) Standard	Tornado Red (Non Metallic) Standard	Urano Grey (Non Metallic) Standard	Pure White (Non Metallic) £260	Carmen Red (Metallic) £540	Limestone Grey (Metallic) £540	Night Blue (Metallic) £540	Pacific Blue (Metallic) £540
----------------------------------	--	---------------------------------------	-----------------------------------	-------------------------------	-----------------------------------	-------------------------------	---------------------------------

Thinking procedurally
Unit 10 Computational thinking

Demonstration

Worksheet 3

- Now try the questions in the worksheet

Save as: S10 C49 Worksheet (your name)
Save in: Section 10

Thinking procedurally
and in computational thinking

Activation

Modular programming

- Modular design and programming techniques are most useful for large, complex programs
- Some programs have thousands or even millions of lines of code
- In small programs of less than a page of code, it may not be worth writing individual modules for every subtask

ostream

```
"Hello, world! \n";
```

Thinking procedurally

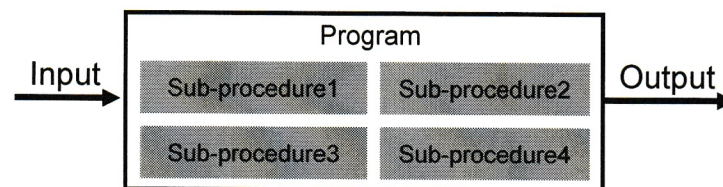
Demonstration

Q2: Draw a hierarchy chart for a program which asks the user which times table they would like to be tested on, and then displays five questions, getting the user's answer each time and telling them whether they were right or wrong. If they are wrong, the correct answer is displayed.

TT ?
|
input
|

Consolidation: Thinking procedurally

- The components of the problem need to be identified
- The components of the solution need to be identified
- The order of the steps needs to be determined
- The necessary sub-procedures must be identified



Homework

1. Write notes on Chapter 49
2. Complete exercises on chapter 49
3. Complete homework worksheet on chapter 49